第 39 卷第 9 期
录用定稿
计算机应用研究
Application Research of Computers
Vol. 39 No. 9
Accepted Paper

一种融合服务聚类与协作相似度的服务替换方法 *

田雨晴，彭　菲，王华东†，胡　强
(青岛科技大学 信息科学技术学院, 山东 青岛 266061)

摘　要：当前服务替换方法多关注于替换服务与失效服务在功能和质量层面的匹配，缺乏对二者协作关系的考量。针对上述问题，本文提出一种融合服务聚类与协作关系的服务替换方法。首先，利用服务之间的组合关系构建服务协作图谱，基于 Node2Vec 进行序列采样并生成协作向量，进而计算服务之间的协作相似度。然后，通过引入服务簇缩减替换时的服务查找空间，从功能和质量层面快速构建候选替换服务集合。最后，在候选替换服务集合中综合服务质量评分和协作相似度实现可替换服务的择优推荐。实验证明本文方法可以有效提高服务替换的效率与合理性。
关键词：服务聚类；协作相似度；服务替换；流程共现
中图分类号：TP　　doi: 10.19734/j.issn.1001-3695.2022.02.0045

Service substitution method combining service clustering and collaboration similarity

Tian Yuqing, Peng Fei, Wang Huadong†, Hu Qiang
(*College of information science & technology*, *Qingdao University of Science & Technology*, *Shandong Qingdao* 266061, *China*)

Abstract: The current service substitution methods mainly focus on the function and quality matching between the substitutive service and invalid one. These methods rarely considered service collaboration. To solve the above problem, this paper proposed a service substitution method integrating service clustering and collaboration. Firstly, this method constructed service collaboration map according to service composition relationships. It used Node2Vec to sample node sequences and generate collaboration vectors from the service collaboration map. This method computed service collaboration similarity based on these collaboration vectors. Then, this paper introduced service clustering to reduce the search space during service substitution. , It generated the candidate substitutive service set from the functional and quality levels. Finally, this method integrated the service quality score and collaboration similarity to achieve the recommendation of substitutive services. Experiments show that the proposed method can effectively improve the efficiency and rationality of service substitution.
Key words: service clustering; collaboration similarity; service replacement; process co-occurrence

0　引言

随着云计算、物联网和 5G 等新一代通信技术的成熟，网络中的服务数量迅速增加，面向服务架构(SOA, Service-oriented Architecture)的软件开发和部署模式得以广泛应用[1]，不同类型的服务被集成到各类业务系统。为了便于调用，服务通常被设计为较小功能粒度的网络应用程序[2]。对于复杂的服务请求，需要采取服务组合的方式将一组功能相关的服务构建为服务流程，通过流程中的服务协同完成响应。

不论是互联网、物联网还是各类自组网络中的服务，均处于动态多变的网络环境中，可能会造成某些服务工作状态的中断[3]。各类服务平台为了提供质量更好、更具备竞争力的服务，采取大量激励机制促使平台中的服务不断进行功能演进和服务版本的更新[4]。因此，不论是服务所处的复杂网络客观环境还是主观层面的功能演化需要，均可能会造成一些服务失效，已有的业务系统面临着服务替换与迁移的需求，如何高效、准确地实现服务替换是采用 SOA 架构进行业务系统开发和部署所面临的一个重要问题。

现有的服务替换研究工作，主要从功能与质量两个层面考虑服务的匹配度，相关研究成果也集中在如何进行服务功能相似度的计算以及质量(QoS, Quality of Service)择优。由于网络中的服务数量众多，已有服务替换方法在进行功能相似服务查找时面临着巨大的查找空间，替换服务的查找过程复杂且耗时较大[5]。此外，现有的服务查找方法缺乏对服务协作因素的考虑，在获取功能层面可以替换失效服务的候选服务集合后，通常选取具备最佳 Qos 的服务作为替换服务。此类做法主要关注了被替换服务位置处的服务质量局部最优化，忽略了替换服务与被替换服务所处业务流程环境中其他服务之间的协作关系，缺乏从全局角度评估替换服务与业务上下文环境中其他服务之间是否具备较高的协作兼容性[6]。

鉴于当前服务替换方法普遍面临着服务查找空间大、查找效率不高和缺乏对替换服务与流程业务上下文协作适应度的考量，本文提出一种融合服务聚类与协作相似度的服务替换方法，文中主要工作和贡献如下：

1) 构建了服务协作图谱，提出一种基于 Node2Vec 的服务协作相似度计算方法，通过引入替换服务与失效服务之间的协作相似度进一步提升了服务替换的合理性。

2) 建立了面向服务簇的服务请求响应模式，通过引入服务簇，缩减服务查找空间，提高了服务替换时的候选服务集合的查找效率。

3)开展了仿真实验，从替换服务的查找时间和替换服务与失效服务所处业务流程中其他服务的共现率两个角度验证

收稿日期：2022-02-06；**修回日期**：2022-03-29　　**基金项目**：国家自然科学基金资助项目(61973180)；山东省自然科学基金资助项目(ZR2019MF033, ZR2021MF092)；山东省重点研发计划软科学项目(2021RKY02037)；云南省教育厅科学研究基金项目(2022J0635)

作者简介：田雨晴(1999-)，女，山西西安人，硕士研究生，主要研究方向为服务计算、人工智能；彭菲(2000-)，女，山东菏泽人，硕士研究生，主要研究方向为服务计算；王华东(1977-)(通信作者)，男，山东青岛人，高级实验师，硕士，主要研究方向为软件工程、大数据技术(whd_1027@163.com)；胡强(1980-)，男，山东邹城人，副教授，硕导，博士，主要研究方向为服务计算、软件工程.

了本文方法的有效性。

1 相关工作

服务替换是服务计算领域中的一个重要研究问题，研究者在进行服务替换时首先考虑替换服务在功能层面与失效服务是否兼容，然后融入服务的质量评价、查找效率、接口数据的一致性、组合偏好等因素，提出了一系列的替换方法。例如，Liang 等使用服务描述各种标记中的术语共现情况对服务进行分类，建立服务操作接口的兼容性和替换的方法，服务替换精确度达到了 85%，但该方法在识别替代服务的语义异构性层面仍面临着很大挑战[7]。Zhang 通过挖掘 Web 服务执行上下文特征规则，定性地表示 Web 服务在不同执行上下文条件下的不同性能，根据当前执行上下文条件，采用基于知识发现的方法选择最佳候选替代服务[8]。Gao 提出了一种数据一致性检查方法，用于动态替换服务，利用数据替换模式来规范不同类型的替换行为，通过实例分析和实验验证了所提方法在动态替换数据一致性方面的有效性[9]。

Santhanam 在进行服务替换时，使用偏好网络表示和推理服务需求对非功能属性的偏好，依据非功能属性进行替换服务选择[10]。Sara 将 Web 服务视为一组操作，服务替换转换为服务操作之间的替换，将已有服务的服务操作构建为一个网络，定义了 4 种服务操作接口匹配度，在需要进行服务替换时，从服务操作形成的网络中进行查找，相比直接进行 Web 服务的替换显著提高了替换效率[11]。Du 等提出了一种基于服务簇网的服务替换方法，通过引入服务簇提高查找效率，但文中的服务簇要求参数接口一致，因此该方法在进行替换服务查找时，服务查找空间缩减空间有限，服务替换的柔性有待进一步提高[12]。Hu 等提出了一种基于流程协作关系的服务替换方法，过引入流程协作提升了替换合理性，但该方法存在流程协作依赖关系采集不完备问题，所计算的服务协作强度精确度有待进一步提高[6]。

服务替换是以失效服务的功能作为服务需求而进行的服务再发现。因此，服务发现领域的一些成果可为服务替换提供借鉴。服务发现的效率与精确度是评价和改进服务发现方法的两个关键指标。研究者通常采用聚类[13, 14]、二分图[15, 16]、索引[17]等方法改进服务发现时的检索机制，提升服务查找速度，其中聚类技术使用最为广泛，它可以有效缩减服务查找空间，提高查找效率。服务发现的精确度主要取决于需求与服务之间的相似度判定。在早期服务发现研究中，通常是从服务描述文档(如 WSDL 文档)中提取若干代表服务特征的关键词，然后通过计算这些关键词的词频[18, 19]或语义相似度[20, 21]来实现需求与服务的相似度判定。

近年来，采用自然语言文本作为服务描述的服务越来越多，基于主题模型或深度学习方法从服务描述文本中提取服务主题特征，并依据主题特征向量进行需求和服务之间的相似度判定成为主流方法。例如，Nabli 等将 TF-IDF 与 LDA 模型相结合，提出一种适用于云服务相似度计算的主题向量生成模型，提高了云服务爬取过程中的发现精确度[22]。胡强等利用 GSDMM 模型生成服务表征向量，并引入概率修正因子，提高了服务发现和聚类过程中的相似度计算精度，并实验证明了在生成服务表征向量时，GSDMM 模型的主题表达能力显著高于 LDA、BTM 等常用主题模型[23]。Lizarralde 使用变分自动编码器从服务描述中学习特征，将编码表示限制为对潜在变量建模，该方法优于 LDA 和 LSA 等传统主题模型，但服务特征学习和模型训练复杂度显著高于常用主题模型[24]。

上述服务发现领域的重要研究成果为服务替换的研究提供了重要的借鉴。目前尚未有研究成果将主题模型引入到服务替换的功能匹配中，为进一步提高服务替换的效率和合理性，本文将服务聚类、GSDMM 主题模型和协作相似度引入到服务替换，通过服务聚类缩减替换过程中的服务查找空间，基于 GSDMM 为服务描述生成主题向量，改进替换过程中的服务相似度计算精度，从而提高替换服务的匹配精确度。通过建模服务之间的流程转移依赖关系，构建一种基于 Node2Vec 的服务协作相似度计算方法，综合考虑服务质量与协作相似度，提升替换服务的推荐合理性。

2 服务协作相似度

社交网络理论研究显示，多名参与者协同完成一项任务时，参与者更倾向与自己熟悉或曾经有过合作经历的人一起分工合作[25]。在服务组合应用场景中，用户也倾向选择有过历史组合关系的服务构建新的业务流程，这些服务在接口兼容、业务部署等层面都具有较高的可靠性。因此，在一组候选替换服务中，选择与失效服务具有相似协作伙伴的服务将增加替换的合理性。

为了度量服务之间是否具备类似的服务协作关系，提出服务协作相似度的概念，利用协作相似度的大小衡量两个服务在相似组合服务流程上下文环境中出现的概率大小。服务协作相似度的大小取决于以下两个要素：服务共现率和流程距离。两个服务在相似的服务流程中出现的次数越多，二者协作相似度越大；两个服务与同一个服务的流程距离越近，二者协作相似度越大[23]。

为了计算两个服务之间的协作相似度，本文构建服务协作图谱。服务协作图谱为一无向图，图谱的节点表示服务，边表示两个节点服务之间具备协作关系。在各类云平台上积累了大量的服务流程模型，通过遍历服务流程模型，将流程模型中的服务抽象为服务协作图中的节点，服务转移依赖关系映射为协作图谱的边即可完成服务协作图谱的构建。

定义 1 协作依赖

在一个服务组合流程模型 sp 中，若服务 s_i 与服务 s_j 存在业务逻辑转移，则称服务 s_i 与 s_j 互为协作依赖关系，记为 $s_i \rightarrow s_j$。

定义 2 服务协作网络

服务协作网络定义为一个无向加权图 $SCG=(V, E)$

a) $V=\{v_1, v_2, v_3,\ldots,v_n\}$为服务节点集合，每个节点 v_i 表示一个服务；

b) $E=\{e=(v_i, v_j)|1\le i, j\le n\}$为协作边集合，其中 $e=(v_i, v_j)$表示服务节点 v_i 与 v_j 所对应的服务 s_i 和 s_j 满足 $s_i \rightarrow s_j$；

图 1 是利用 ProgrammableWeb 网站中注册的 Mashup 服务所构建的服务协作图谱片段。Mashup 服务是一种组合服务，一个 Mashup 服务通常包含 2 到 6 个左右的 Web 服务，这些 Web 服务协同完成复杂的业务功能。在构建了服务协作图谱后，采用 Node2Vec 对图谱中的节点进行序列采样，并根据采集到的协作序列为每个节点所代表的服务生成向量，将该向量称为服务协作向量。

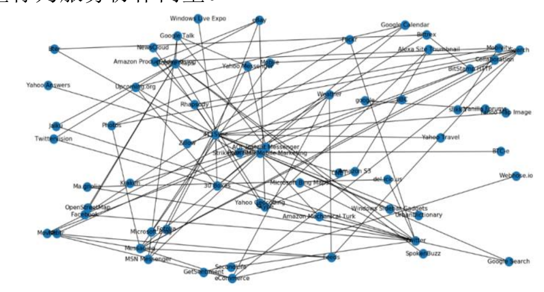


图 1 服务协作图谱示例片段

Fig. 1 A sample fragment of service collaboration graph

算法 1 用于生成服务协作相似度矩阵，在该矩阵中可以获得任意两个服务之间的协作相似度。算法 1)-6)行用于构建协作图谱，首先初始化一个空协作图谱 SCG，然后遍历流程库 SP 中

的服务流程，将存在协作依赖关系的服务 s 与 s' 作为节点 $v(s)$ 和 $v(s')$ 加入 SCG，并在两个节点之间构建一条边<$v(s)$, $v(s')$>。

算法 7)-20)行采用 Node2Vec 生成服务协作向量[26]。算法在第 7)行初始化空集合 *walks* 用来存放随机游走得到的节点序列。算法第 8)行通过循环产生 r 次随机游走。对于服务协作图谱中的每个服务节点 u，算法第 9)和 10)行为其初始一个序列数组 *walk*[*u*]。算法第 11)至 17)行每次游走生成长度为 l 的节点序列存放于 *walk* 中，主要通过等概率随机选择邻居节点和负采样的方式生成节点序列，并将 *walk* 添加到 *walks* 中。第 19 行)采用 *SGD* 方法对 *walks* 进行训练，为每个节点生成服务协作向量 *scv*[i]。算法 21)-23)行，采用向量的余弦夹角值来度量任意两个服务之间的协作相似度，并构建协作相似度矩阵 *CSM*，算法 24)行返回构建的协作相似度矩阵 *CSM*。

算法 1　Collaboration_SimCalculate

```
Input: the set of service process SP;
Output: collaboration similarity matrix CSM
1) Initialize a service collaboration graph SCG;
2) for each service process sp in SP
3)     for each service s∈sp
4)       if (∃s'∈sp, s.t. s→ s'∨ s'→s) add v(s), v(s') and <v(s), v(s')> into SCG.
5)     endfor
6) endfor
7) Initialize walks to Empty
8) for iter = 1 to r do
9)     for all nodes u ∈ V do
10)         Initialize walk to [u]
11)         for walk_iter = 1 to l do
12)             curr = walk[-1]
13)             V_curr = GetNeighbors(curr, SCG)
14)             t= AliasSample(V_cur, π)
15)             Append t to walk
16)         endfor
17)         Append walk to walks
18)     endfor
19)     scv[v_i]= StochasticGradientDescent(k, d, walks)
20) endfor
21) for ∀v_i, v_j in SCG
22)     CSM[v_i][v_j]= Cosine_Vec(scv[v_i], scv[v_j])
23) endfor
24) return CSM.
```

3　面向服务簇融合流程协作相似度的服务替换

3.1　面向服簇的服务请求响应模式

传统的 SOA 架构中，服务请求由发布在网络中的服务或一组服务的组合流程实现响应。服务查找过程中面临着海量服务的比对，服务查找效率不高。服务替换时需要重新为已有响应需求的服务寻找一个替代服务，在这种模式下，可替代服务的查找代价巨大。

为了提高服务替换时的服务查找效率，在 SOA 架构中引入服务簇，建立了如图 2 所示的服务响应模式。在传统的 SOA 请求响应模式中，增加一个虚拟资源层，在虚拟资源层中构建服务簇。不同于传统服务请求响应模式，位于业务模型层中的服务请求不再由原子服务响应，而是首先由虚拟资源层中的服务簇响应，然后从服务簇中绑定最优响应服务，进而绑定物理资源层中原子服务或组合服务流程。

引入虚拟资源层和服务簇，可以提高服务请求的响应粒度，在服务发现时采用“服务簇-服务”二级查找模式，由于将相似的服务划分为一个簇，因此大幅度缩减了服务查找空间，提高服务查找效率。

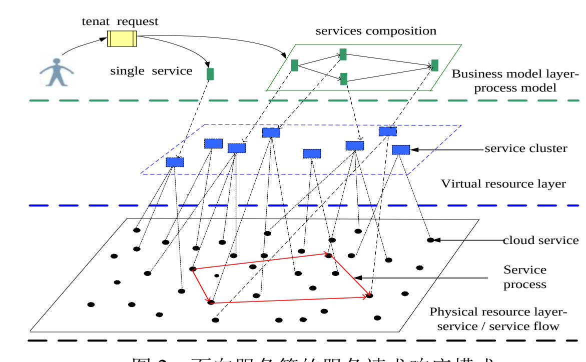


图 2　面向服务簇的服务请求响应模式

Fig. 2　Service request and response schema oriented service clusters

3.2　服务替换算法

服务替换过程划分为两个阶段，第一个阶段为候选服务集合的生成，该阶段从功能层面获取可以替换失效服务的服务集合；第二个阶段进行服务质量和协作相似度的综合择优。为了便于描述簇模式下的服务替换过程，首先给出服务及服务簇的形式化定义。

定义 3　服务

服务定义为 6 元组 $s=(Id, D, I, O, Q, L)$,其中：Id 为服务的标识，D 为服务的功能描述文本，I 和 O 分别是输入和输出参数集合，Q 是质量集合，L 为服务的地址标识。

例如，ProgrammableWeb 上注册的 Google 地图服务中的 GeocodingApi 服务，其对应的服务描述 D 为“Geocoding is the process of converting addresses (like "1600 Amphitheatre Parkway, Mountain View, CA") into geographic coordinates (like latitude 37.423021 and longitude -122.083739), which you can use to place markers or position the map. Reverse geocoding is the process of converting geographic coordinates into a human-readable address.”。该服务的输入参数集合 I={(string address)}，输出参数集合 O={(float latitude), (float longitude)}。

服务质量集合 $Q=\{q_i\}$, $q_i=\{N,C,V,U\}$，其中 N 为服务质量参数名称，C 是比较运算符，V 是参数的值，U 为其单位。如果 $q=(ResponseTime,<,0.5,ms)$，表示服务响应时间低于 0.5 毫秒。

定义 4　服务簇

服务簇定义为 7 元组 $sc=(C_{Id}, V, I, O, S_w, Q_c)$，其中 C_{Id} 是服务簇的标识，V 为服务簇的功能向量，I 和 O 分别为输入和输出参数集合，S_w 是服务簇所包含的服务集合，Q_c 为质量参数集合。

服务簇由若干服务聚类得到，不设置功能描述文本，在定义中采用功能向量 V 来表示服务簇的功能。服务簇的功能向量 V 为簇中所有服务的功能向量均值。服务簇的输入和输出参数是其所包含的所有服务的输入输出参数集合的并集。服务簇的质量参数值应该能涵盖所有服务的质量参数取值，采用区间法 $q_i=\{N,C,[Vmin, Vmax],U\}$ 表示服务簇的质量参数值。若服务簇的 $Q_c=\{(ResponseTime, <, [0.1,0.5], ms), (ResponseRat, >, [98,100], \%)\}$，则表示该服务簇中的服务响应时间最低小于 0.1*ms*，最大小于 0.5*ms*，而服务响应率最低的服务大于 98%，最高的服务为 100%。

本文采用适合短文本主题提取的 GSDMM 模型为服务描述文本生成功能向量。在前期工作中验证了 GSDMM 为 Web 服务描述生成的功能向量的质量显著高于其他主题模型[23]。GSDMM 基于狄利克雷混合模型(DMM)生成文档，然后使用吉布斯采样(Gibbs Sampling)算法近似求解模型。DMM 如图 3 所示，吉布斯采样过程为使用一个单词在所有主题上不断地采样，最终得到这个单词的主题分布矩阵，从而得到文档-主题矩阵 $\Theta=d\times z$ 以及单词-主题矩阵 $\Phi=w\times z$。吉布斯

采样中一篇描述属于某个主题的概率计算公式如下式(1)：

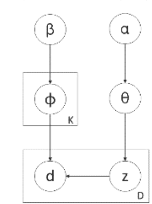


图 3　DMM 模型

Fig. 3　DMM model

$$p\left(z_d = z \mid \vec{z}_{\neg d}, \vec{d}\right) \propto \frac{m_{z,\neg d}+\alpha}{D-1+K\alpha} \frac{\prod_{w\in d}\prod_{j=1}^{N_d^w}\left(n_{z,\neg d}^{w}+\beta+j-1\right)}{\prod_{i=1}^{N_d}\left(n_{z,\neg d}^{w}+V\beta+i-1\right)} \qquad (1)$$

其中，K 表示初始主题个数，D 表示语料库中描述总数， m_z 表示主题 Z 下的文档数，n_z 表示主题 Z 下的单词数， $n_{z,\neg d}^{w}$ 表示主题 Z 下的单词 w 出现的次数， $\neg d$ 表示去除当前文档。

定义 5　子集参数

Pa 和 Pb 是两组参数，Pa 称为 Pb 的子集参数当且仅当：

a) Num(Pa)≤Num(Pb)，Num(p)表示参数 p 的个数；

b) ∀mi∈Pa, ∃nj∈Pb，满足 mi∝nj 且 Type(mi)≅Type(nj)；

Type(m)表示参 m 的类型，符号∝表示语义等价，≅表示参数类型兼容，如果 Pa 是 Pb 的子集参数，记为 Pa⇐Pb；

在进行服务替换时，首先需要从功能和接口层面寻找合适的候选替换服务，然后再从服务质量和协作相似度角度计算最佳替换服务。本文采用服务的描述文本的功能相似度作为两个服务的功能相似度。对于服务 s_1 和 s_2，$v(s_i)$表示基于功能描述文本 $s_i.D$ 和 GSDMM 模型为其生成的功能向量(i=1, 2)，则服务的功能相似度为两个向量的余弦夹角值，参见式(2)：

$$FucSim(s_1, s_2)=\frac{v(s_1)\bullet v(s_2)}{|v(s_1)|*|v(s_2)|} \qquad (2)$$

在对服务质量进行量化时，假设 $S=\{s_1, s_2, \ldots, s_m\}$为一组候选服务，每个服务具有 n 个质量参数，$[q_{i1}, q_{i2}, \ldots, q_{in}]$是服务 s_i 的质量参数值，则采用式(3)对 s_i 的服务质量进行加权评分量化，参数ω_j 用于表示各个质量参数的权重。

$$Qscore(s_i)=\sum_{j=1}^{n} q_{ij} * w_j, \quad \sum_{j=1}^{n} w_j = 1 \qquad (3)$$

$$q_{ij}=\begin{cases}\dfrac{q_j^{\max}-q_{ij}}{q_j^{\max}-q_j^{\min}}, & if\ q_j^{\max}-q_j^{\min}\neq 0;\\ 1, & if\ q_j^{\max}-q_j^{\min}=0;\end{cases} \qquad (4)$$

$$q_{ij}=\begin{cases}\dfrac{q_{ij}-q_j^{\min}}{q_j^{\max}-q_j^{\min}}, & if\ q_j^{\max}-q_j^{\min}\neq 0;\\ 1, & if\ q_j^{\max}-q_j^{\min}=0;\end{cases} \qquad (5)$$

在对服务质量进行量化时，式(4)用于负向质量参数的量化，例如响应时间、价格，此类服务的数值越小表示服务的质量越高。式(5)用于正向质量参数的量化，例如稳定性、吞吐量，此类服务的数值越大表示服务的质量越高。

算法 2 为本文所构建的服务簇模式下融合服务协作相似度的服务替换算法。算法第 1)行首先初始化两个空集合 *CSR* 和 *cs_r*，分别用于存储候选服务簇和候选替换服务集合。算法 2)-4)行利用功能相似度 *FucSim* 筛选存在可替换失效服务 *se* 的服务簇，将与失效服务 *se* 功能等价的服务簇加入到集合 *CSR*。算法 5)至 9)行，遍历 *CSR* 中所有服务簇所包含的服务，利用子集参数的定义规则，将接口参数可与失效服务 *se* 接口参数匹配的服务加入到候选服务集合 *cs_r*。

算法 10)-12)行用于计算候选替换服务集合 *cs_r* 中每个服务的推荐评分。服务 *s* 的推荐评分由服务的质量评分和服务 *s* 与失效服务 *se* 之间的协作相似度加权得到，在算法的使用过程中，依据服务质量和协作相似度对替换服务推荐的重要度不同，设置合理的α和β值即可。算法第 13)行通过计算服务的推荐评分，选择 *cs_r* 中评分最高的服务作为替换服务 *rst*，算法第 14)行最终返回推荐服务 *rst*。

算法 2　Service_Subtitution_Collaboartion

```
Input: the cloud service cluster pool CSCP; the invalid
cloud service se;
Output: the recommended substitution service set rst;
1) CSR=∅, cs_r =∅;
2)  for each sc∈CSCP
3)    if (FucSim(sc, se)>δ)  CSR = CSR ∪{sc};
4)  endfor
5)  for each sc∈CSR
6)     for each s∈sc.Sw
7)        if (s.I⇐se.I∧se.O⇐s.O)  cs_r=cs_r ∪{s};
8)     endfor
9)  endfor
10) for each service s in cs_r
11)  RecomGrade(s)=α*Qscore(s)+β*CSM(s, se);
12) endfor
13) rst={s| max(RecomGrade(s))∧s∈cs_r};
14) return(rst).
```

4　仿真实验

本节通过实验验证所提出服务替换方法的有效性。实验分为两个部分，一部分用于验证本文方法是否提高了服务替换时的查找效率，另外一部分用于验证所推荐的替换服务是否较其他方法更为合理。实验采用的计算机软硬件配置为：CPU i5-10210U，内存 8G，Windows 10 操作系统，采用 python 编写验证程序。

4.1　实验数据集

实验中构建了两个数据集：数据集 1 为真实数据集，从 ProgrammableWeb 网站中爬取 Mashup 服务，经过数据清洗保留 5327 条有效 Mashup 服务，共计包含 786 个 WebAPI 服务，根据服务标签这些服务被聚类为 34 个服务簇。为了提高服务替换的可行性，便于服务接口匹配，手工对 786 个服务中的输入输出参数进行了修正，实现同义数据类型和参数名词的统一表示。

数据集 2 为仿真数据集，该数据从 ProgrammableWeb 网站中爬取的 156 个类别、共计 21425 个有效 Web API 服务中，随机抽取了 80 个类别，每个类别随机抽取 1-10 条服务，共计 400 条 Web API 服务。将此 400 个服务称为原始服务，按如下流程进行处理，生成 3600 个新服务，共计 4000 个服务。

1)服务描述的生成

对于每一条原始 Web API 服务对应的服务描述，随机生成 1-20 条服务描述，每条新服务描述由原服务描述文本采取按照比例从 10-70%随机替换单词的方式生成。

2)服务接口参数的生成

对于每一条原始 Web API 服务的参数，获取该参数名称的 2 个同义词，与原参数一起作为候选参数名称集合，在由原始服务生成新服务时，随机在候选参数名称集合中选择一个参数名称作为新服务的参数。

3)服务质量的设置

由于在 ProgrammableWeb 上无法获取到 Web API 服务的质量参数信息，本实验中选取服务响应时间 rt，可靠性 av 和费用 cs 三个质量参数，为数据集 1 和数据集 2 中的每一条服务在下述取值区间内{rt[0.01, 0.1]，av[97%, 100%], cs[0, 20]}随机生成服务质量的参数值。

4)协作关系的生成

随机生成 40000 条包含 8 个服务节点的服务流程，将

4000 条服务按照类别划分为 8 类，每类大约 500 条服务，对应于服务流程的一个节点。在生成服务流程时，随机从对应的一类服务中抽取 1 条服务形成一个节点，从 8 类中抽取 8 条服务组成一条服务流程。

4.2　服务查找与替换效率

服务替换的本质是依据失效服务的功能重新查找一个替代服务，因此，服务的查找和替换效率是一个重要的验证指标。选取了近年来发表的四种服务查找(文献[22]、文献[27]、文献[28]和文献[29])和替换方法(文献[6]、文献[11]、文献[12]和文献[30])与本文方法(SM_CC)进行对比验证。

对于数据集 1，本文采取数据中服务的类别数 34 作为聚类数量。对于数据集 2，分别将其聚类为 40，80，120，160 和 200，不同类别下服务查找耗时折线图参见图 4，从图中可以看出，服务查找效率和服务簇的大小有关，随着服务簇粒度的变小，查找耗时呈现先降低后增加的趋势，依据折线趋势和耗时数据，本文选取服务查找速度最快的类别数目 120 作为聚类的数量。

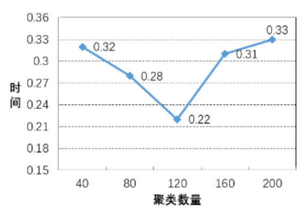


图 4　数据集 2 的聚类数量与查找时间

Fig. 4　Discovery time for different service cluster number in dataset 2

图 5 和图 6 分别为不同方法在两个数据集中查找服务的耗时对比图。从图 5 和图 6 中的柱状图对比可以看出，本文方法的查找效率高于其他四种方法，在数据集中的服务数量较大时，服务簇的引入缩减查找空间更为明显，服务查找耗时相比其他方法更低，在数据 1 中，本文的方法与其他四个方法的平均耗时的占比为 1:1.79，在数据集 2 中，随着服务数量的增加，上述耗时占比变为 1:1.96，耗时平均降低 8.67%。此外，通过引入主题模型对服务功能向量进行预训练的方法，在服务查找时效率要高于其他方法。

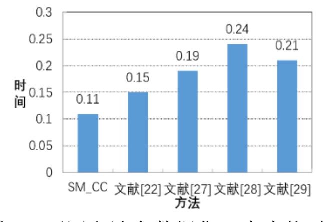


图 5　不同方法在数据集 1 中查找时间

Fig. 5　Discovery time for different methods in dataset 1

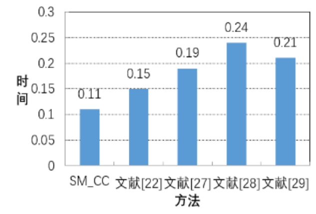


图 6　不同方法在数据集 2 中查找时间

Fig. 6　Discovery time for different methods in dataset 2

图 7 和图 8 分别为不同服务替换方法在两个数据集中的替换耗时对比图。从图中数据可知，本文方法在服务替换时，耗时低于其他服务替换方法。在五种方法的对比中，服务替换时引入服务簇的方法，在替换时耗时低于未引入服务簇的方法。在所有替换方法中，Sara 所提出的方法因构建了复杂的接口比对操作而耗时最高。在数据 1 中，本文方法与其他四种替换方法平均耗时比为 1:1.75，上述耗时占比在数据集 2 中为 1:1.93，本文方法在替换效率上显著优于其他方法，从数据 1 与数据 2 的耗时占比看，随着服务数量的增加，替换耗时占比平均降低了 9.4%，进一步印证了文本方法引入服务簇后显著提高了服务查找与替换效率。

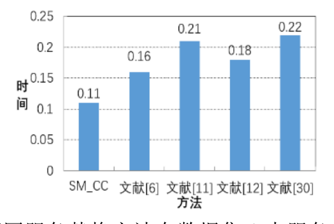


图 7　不同服务替换方法在数据集 1 中服务替换耗时

Fig. 7　Time-consuming for different service substitution methods in dataset 1

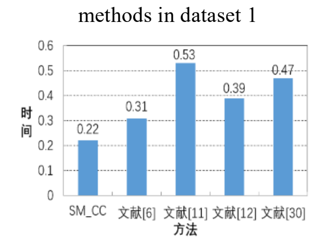


图 8　不同服务替换方法在数据集 2 中服务替换耗时

Fig. 8　Time-consuming for different service substitution methods in dataset 2

4.3　替换的合理性

在保证服务在功能层面能够满足失效服务的前提下，文中的算法综合了服务质量和流程协同效应等两个方面因素推荐最优替换服务。本文提出服务共现率的概念，利用推荐服务与失效服务所处的业务流程上下文中的服务共现概率来评价替换服务的合理性。

服务的共现率是两个服务在同一个流程中出现的次数与二者在所有两次中出现次数之比。令 OccuNum(s_i)表示 s_i 在所有服务组合流程中出现的次数，则服务 s_i 与 s_j 的共现率可以通过公式 ServiceCo_Occu(s_i, s_j)= OccuNum($s_i \cap s_j$)/(OccuNum(s_i)+ OccuNum(s_j))。将替换服务与失效服务的流程共现率定义为替换服务与失效服务所在流程中所有其他服务的共现率之和。假设失效服务 *se* 所处的业务流程中的服务集合为 *S*，替换服务为 *st*，则 *st* 与 *se* 的服务流程共现率定义为 Proc_SerCo_(*st*, *se*)= ServiceCo_Occu(*S*-*se*, *st*)。

图 9 和图 10 分别展示了在数据集 1 和数据集 2 中，不同替换算法推荐服务与失效服务的服务流程共现率。从图中数据可以看出，在两个数据集中，本文方法所推荐的替换服务的流程共现率均高于其他方法，从数值上高出大约 2-6 倍，从而验证了本文方法将服务协作相似度加入到服务替换过程中，提高了服务替换的合理性。

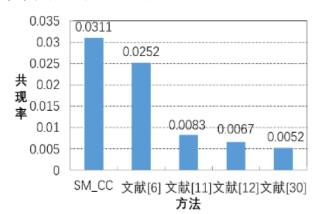


图 9　数据集 1 中各方法的流程共现率

Fig. 9　Process co-occurrence rates for different methods in dataset 1

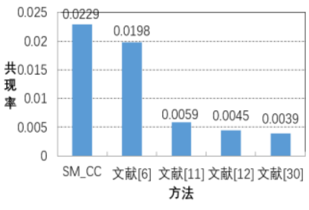


图 10　数据集 2 中各方法的流程共现率对比

Fig. 10　Process co-occurrence rates for different methods in dataset 2

5 结束语

为了提高服务替换的效率和合理性，本文提出一种服务簇模式下融合流程协作相似度的服务替换方法。通过引入服务簇，缩减了服务查找空间，提高了服务替换时的服务发现效率。将服务之间的流程协作关系进行建模和度量，并与服务质量一起引入到替换服务的择优推荐中，提高了服务替换的合理性。实验结果验证了本文方法有效提高了服务替换的效率和合理性。

在未来工作中，将研究聚类粒度大小对服务替换效率的影响，并改进服务协同效应的建模和度量方法以进一步提升替换的效率与合理性。

参考文献：

[1] Akbar M A, Khan A A, Mahmood S, *et al.* A robust framework for cloud‐based software development outsourcing factors using analytical hierarchy process [J]. Journal of Software: Evolution and Process, 2021, 33 (2): e2275.

[2] Cheng H, Zhong M, Wang J. Diversified keyword search based web service composition [J]. Journal of Systems and Software, 2020, 163: 110540.

[3] Sefati S, Navimipour N J. A QoS-Aware Service Composition Mechanism in the Internet of Things Using a Hidden-Markov-Model-Based Optimization Algorithm [J]. IEEE Internet of Things Journal, 2021, 8 (20): 15620-15627.

[4] 王先清，黄昌勤，罗旋，等. 基于语义扩展类型论的云服务替换性判定研究 [J]. 通信学报，2016, 37 (2): 20-31. (Wang Xianqing, Huang Changqin, Luo Xuan, *et al.* Determining substitutability of cloud services supported by semantically extended type theory [J]. Journal on Communications, 2016, 37 (2): 20-31.)

[5] Al-Sayed M M, Hassan H A, Omara F A. An intelligent cloud service discovery framework [J]. Future Generation Computer Systems, 2020, 106: 438-466.

[6] Hu Q, Shen J. A Cluster and Process Collaboration-Aware Method to Achieve Service Substitution in Cloud Service Processes [J]. Scientific Programming, 2020: 1298513.

[7] Liang Q, Lee B S, Hung P C K. A rule-based approach for availability of service by automated service substitution. [M]// Software: Practice and Experience, 2014, 44 (1): 47-76.

[8] Zhang M W, Zhu Z L, Li D C, *et al.* An execution context aware approach for Web service substitution [J]. In 7th International Conference on Advanced Information Management and Service. IEEE, 2011: 13-18.

[9] Gao H, Duan Y, Miao H, *et al.* An approach to data consistency checking for the dynamic replacement of service process [J]. IEEE Access, 2017, 5: 11700-11711.

[10] Santhanam G R, Basu S, Honavar V. In IEEE International Conference on Services Computing [C]// IEEE, 2009: 210-217.

[11] R. Sara, A. Bakhta, and L. Lakhdar, A similarity network for web services operations substitution [J]. Journal of King Saud University-Computer and Information Sciences, 2020, 32 (9): 1055-1062.

[12] Du Y Y, Gai J J, Zhou M C. A Web service substitution method based on service cluster nets [J]. Enterprise Information Systems, 2017, 11 (10): 1535-1551.

[13] Natarajan B, Obaidat M S, Sadoun B, *et al.* New clustering-based semantic service selection and user preferential model [J]. IEEE Systems Journal, 2020, 15 (4): 4980-4988.

[14] Obidallah W J, Raahemi B, Ruhi U. Clustering and association rules for web service discovery and recommendation: a systematic literature review [J]. SN Computer Science, 2020, 1 (1): 1-33.

[15] Chen K, Kuang C. Web service discovery based on maximum weighted bipartite graphs [J]. Computer Communications, 2021, 171: 54-60.

[16] Hafsi A, Gamha Y, Ben Njima C, *et al.* BIG-SWSDM: BIpartite graph based social web service discovery model [C]// International conference on business information systems. Springer, Cham, 2020: 307-318.

[17] Abdelli A, Serrai W, Mokdad L. A novel and efficient index based web service discovery approach [J]. Computer Standards & Interfaces, 2022, 80: 103586.

[18] Cassar G, Barnaghi P, Moessner K. Probabilistic matchmaking methods for automated service discovery [J]. IEEE Transactions on Services Computing, 2013, 7 (4): 654-666.

[19] Agarwal N, Sikka G, Awasthi L K. Enhancing web service clustering using Length Feature Weight Method for service description document vector space representation [J]. Expert Systems with Applications, 2020, 161: 113682.

[20] Cheng B, Li C, Zhao S, *et al.* Semantics mining & indexing-based rapid web services discovery framework [J]. IEEE Transactions on Services Computing, 2021, 14 (3): 864-875.

[21] Abid A, Rouached M, Messai N. Semantic web service composition using semantic similarity measures and formal concept analysis [J]. Multimedia Tools and Applications, 2020, 79 (9): 6569-6597.

[22] Nabli H, Djemaa R B, Amor I A B. Efficient cloud service discovery approach based on LDA topic modeling [J]. Journal of Systems and Software, 2018, 146: 233-248.

[23] Hu Q, Shen J, Wang K, *et al.* A Web service clustering method based on topic enhanced Gibbs sampling algorithm for the Dirichlet Multinomial Mixture model and service collaboration graph [J]. Information Sciences, 2022, 586: 239-260.

[24] Lizarralde I, Mateos C, Zunino A, *et al.* Discovering web services in social web service repositories using deep variational autoencoders [J]. Information Processing & Management, 2020, 57 (4): 102231.

[25] Dania A, Griffin L L. Using social network theory to explore a participatory action research collaboration through social media [J]. Qualitative research in sport, exercise and health, 2021, 13 (1): 41-58.

[26] Grover A, Leskovec J. node2vec: Scalable feature learning for networks [C]// Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 2016: 855-864.

[27] Lu C. A Novel Web Service Discovery Method Combining Semantic Interface Similarity and Context Similarity [C]// IOP Conference Series: Earth and Environmental Science. IOP Publishing, 2021, 693 (1): 012006.

[28] [28Zhang F, Zeng Q, Duan H, *et al.* Composition context-based web services similarity measure [J]. IEEE Access, 2019, 7: 65195-65206.

[29] Abdelli A, Serrai W, Mokdad L. A novel and efficient index based web service discovery approach [J]. Computer Standards & Interfaces, 2022, 80: 103586.

[30] 张朝阳. 面向群组服务的替换方法研究 [D]. 南京：南京邮电大学，2018. 17, 5: 11700-11711. (Zhang zhaoyang. Research on service substitution to group services [D]. Nanjing University of Posts and Telecommunications, 2018. 17, 5: 11700-11711.)